HTTP based Denial of Service Prevention

Sam Smith, Gates Lamb, Yifei Sun

Problem

- DoS and DDoS more easily accomplished
- HTTP uses more compute than TCP or UDP
- Distributed attacks are harder to track
- Constant game of whack-a-mole

Our Solution

- Python HTTP handler
- Two second stateful approach
- Check number of bytes in three headers (‘user-agent’, ‘sec-ch-ua’, and ‘accept’)
- If more than 50 requests with same header size in each state, then block
- Assume the browser has our JavaScript to attach headers

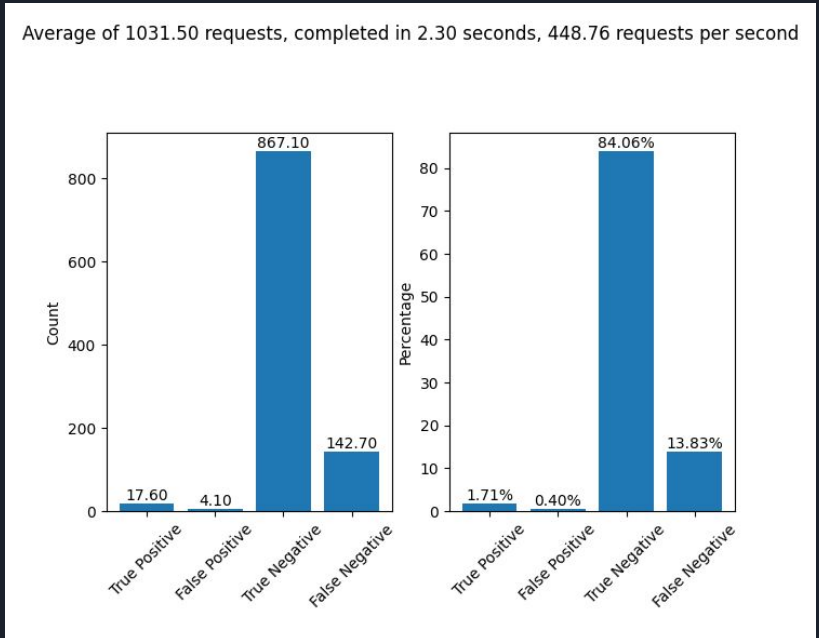
Related Work/Solutions

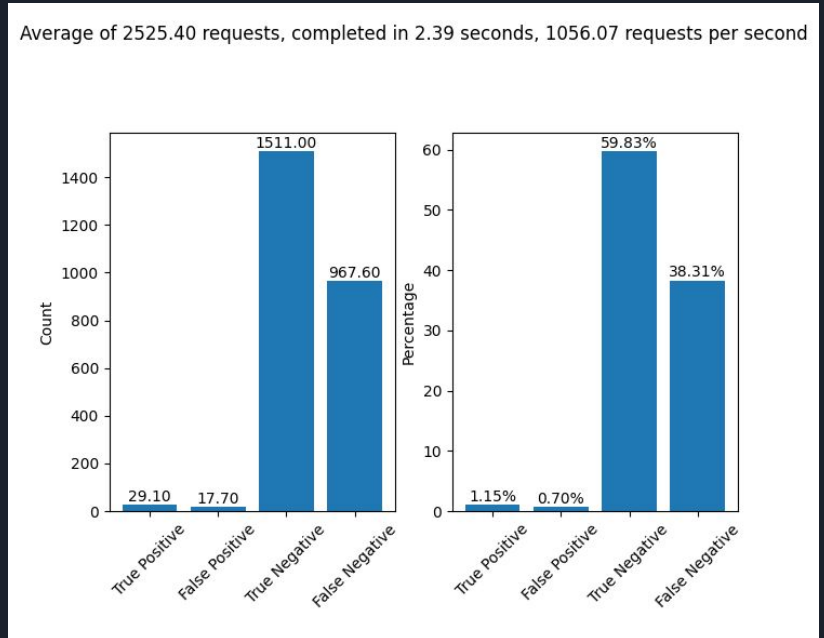
- Traffic Profiling
- IP Reputation
- Security Challenges(CAPTCHA)
- Rate-Limiting
- WAFs
- NIDS/NIPS
    - Previous threat signatures and client classification/behaviour analysis.

Adversary Model

- Sends large amounts of HTTP GET and POST requests to target server
- Uses an array of compromised machines to send traffic from different IPs
- Targets endpoints that require more resources on the server
- Persists over a long period of time

Performance Metrics





Performance Metrics

~1000 requests at ~450 requests per second

- Avg Accuracy: 85.8%
- Avg Detection Rate: 11%
- Avg False Positive Rate: 0.5%

~2500 requests at ~1050 requests per second

- Avg Accuracy: 61%
- Avg Detection Rate: 3%
- Avg False Positive Rate: 1.2%

Results

- Low blocking overall
- Low false positive Rate but also low detection rate.
- Performance metrics suffer as number of requests increases
- Metrics for high number of requests could be improved with blacklist resetting or removing from it over time

Demo